基于自适应吸引半径的萤火虫算法的粒子滤波 *

王航星 1a, 1b，潘　巍 1a, 1b

(1. 首都师范大学 a. 信息工程学院; b. 北京成像技术高精尖中心, 北京 100048)

摘　要：针对粒子滤波算法对粒子数目的大量需求等弊端，提出一种基于改进的萤火虫算法的粒子滤波。首先，在萤火虫的亮度公式中引入观测值信息，以提高算法跟踪的准确性；其次，提出自适应吸引半径参数来控制萤火虫群寻优时的吸引范围，使算法的实时性更好；最终利用萤火虫算法的迭代寻优来进行粒子更新。对比实验表明，该算法在跟踪精度和运行时间上都有所优化，说明该算法即使在粒子数目较少的条件下，也能保证目标跟踪的准确性和实时性。

关键词：自适应吸引半径；迭代寻优；目标跟踪；粒子多样性；相对亮度

中图分类号：TP391.4　　doi: 10.3969/j.issn.1001-3695.2018.06.0397

Particle filter based on firefly algorithm with adaptive attraction radius

Wang Hangxing[1a, 1b], Pan Wei[1a, 1b]

(1. *a. College of Information Engineering*, *b. Beijing Advanced Innovation Center for Imaging Technology*, *Capital Normal University*, *Beijing* 100048, *China*)

Abstract: For the large demand of particles when particle filter tracking, this paper proposed a particle filter algorithm based on the improved firefly algorithm. Firstly, in order to improve the accuracy of tracking, the improved algorithm introduced the observation value into the brightness formula of the firefly; Secondly, when the fireflies swarm optimized, an adaptive attraction radius parameter was proposed to control the attraction range, so that the real-time performance would be better; Finally, the iterative optimization of the firefly algorithm was used to update the particles. Comparison experiments show that, the algorithm has been optimized for tracking accuracy and running time. It shows that the algorithm can guarantee the accuracy and real-time performance of the target tracking even if the number of particles is small.

Key words: adaptive attracting radius; iterative optimization; target tracking; particle diversity; relative brightness

0　引言

人们在生活中所获取到的信息超过 80%来源于视觉，随着计算机视觉领域的飞速发展，基于视频序列的目标跟踪作为计算机视觉应用中的重要任务，被广泛应用在视频监控、车辆跟踪、手势识别等众多领域[1,2]。

传统的目标跟踪主要是采用基于模型线性化和高斯假设的卡尔曼滤波来完成，但在实际应用中，会存在大量的非线性系统或非高斯噪声系统，卡尔曼滤波显然已经满足不了系统要求[3]。基于蒙特卡罗方法的粒子滤波[4]由于在跟踪过程中不需要做线性高斯的假设，可以很好地求解非线性非高斯系统下的跟踪问题，因此得到了广泛的应用[5~7]。

然而，尽管粒子滤波在目标跟踪方面有其一定的优越性，例如可以利用蒙特卡罗方法将积分运算转换为求和运算等，但它存在着一些缺点，如需要大量的粒子来保证跟踪的准确性导致算法实时性差，重采样后造成的粒子贫化问题。因此，基于群智能优化算法的粒子滤波是一个崭新的发展方向。这类算法通过对粒子分布进行迭代寻优减少对粒子数的需求，同时算法不涉及对低权值粒子的舍弃，从根本上解决了粒子贫化问题。

张才千等人[8]利用粒子群算法(PSO)在重采样前对粒子集进行优化，使粒子尽可能多地分布在高似然区域，有效地防止了重采样后粒子多样性的降低。Wang 和 Heris 等人[9,10]分别在重采样阶段和初始化采样阶段使用蚁群算法来优化粒子滤波进行目标跟踪，虽然很好地解决了粒子匮乏及粒子退化的问题，但由于搜索过程和结束条件带有很强的经验性，该方法并没有得到很好的使用。Wang 等人[11]使用改进的遗传算法作为重采样方法，设计交叉变异概率随粒子的退化程度成正比变化来控制交叉变异操作采样新的粒子，虽然提高了粒子的有效性和多样性，但其迭代次数和阈值不易确定。

相比于以上几种算法，萤火虫算法作为最新的群智能优化

收稿日期：2018-06-13；**修回日期：**2018-07-27　**基金项目：**复杂路况下自动驾驶传感器及关键技术研究（61772351）；面向协作机器人的物理性人机安全交互关键技术研究（61702348）

作者简介：王航星（1996-），女，河北邯郸人，硕士研究生，主要研究方向为图像处理、目标跟踪（star_wanghx@163.com）；潘巍（1973-），女，湖北武人，副教授，硕导，博士，主要研究方向为模式识别、智能控制、信息融合.

算法之一，具有较高的收敛精度、更快的收敛速度且容易实现的优点，但目前国内外关于萤火虫算法优化粒子滤波的工作还很少。朱文超等人[12]根据权值的退化程度将粒子集分层并将优化层粒子映射到高似然区域，但不能真正解决粒子贫化问题。Gao 等人[13]实现了改进的吸引度公式的萤火虫算法，使粒子集分布到高似然区域来优化粒子滤波，但亮度计算公式的特性会导致很高的运算复杂度。田梦楚等人[14]提出萤火虫算法优化粒子滤波(FA-PF)，使用改进的位置更新公式进行迭代寻优，避免了在寻优过程中出现局部极值的现象，同时也有效地控制了粒子多样性的衰减，但由于粒子相互吸引时需要全部粒子之间两两进行交互运算，所以会造成计算复杂度的增加和运行时间的浪费。

针对上述问题，本文提出了一种基于自适应吸引半径的萤火虫算法的粒子滤波，通过加入自适应吸引半径参数来控制粒子相互吸引时的吸引范围，解决了粒子交互时参与运算的粒子数目较大以及计算复杂度增加的问题。

1　粒子滤波算法

粒子滤波算法的关键是使用一组带权值的粒子集以及带有噪声的观测值来估计系统状态的后验密度函数[15]，从而实现对系统状态的估计[16]。粒子的运动状态是一阶自回归状态空间模型[17]，状态模型和观测模型数学表示如下：

$$x_k = \mathrm{f}\left(x_{k-1}, \mathrm{u}_{k-1}\right) \tag{1}$$

$$y_k = \mathrm{h}\left(x_k, v_k\right) \tag{2}$$

其中：f 和 h 分别为状态函数和观测函数；u_{k-1} 和 v_k 分别为系统噪声和观测噪声。状态模型和观测模型的动态表示[18]如图 1 所示。

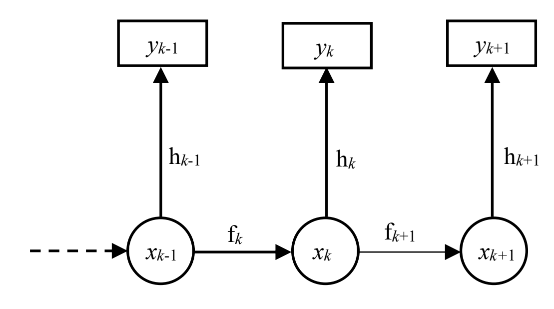


图 1　状态模型和观测模型的动态系统

在已知 $k-1$ 时刻的粒子状态条件下，根据状态模型、观测模型以及贝叶斯估计理论对 k 时刻的粒子状态进行预测和更新[19]。粒子状态预测的计算如下：

$$\begin{aligned} P\left(x_k \mid y_{1:k-1}\right) &= \int \frac{P(x_k \mid x_{k-1}, y_{1:k-1}) P\left(x_{k-1}, y_{1:k-1}\right)}{P(\mathrm{y}_{1:k-1})} dx_{k-1} \\ &= \int P(x_k \mid x_{k-1}, y_{1:k-1}) P\left(x_{k-1}, y_{1:k-1}\right) dx_{k-1} \\ &= \int P\left(x_k \mid x_{k-1}\right) P\left(x_{k-1} \mid y_{1:k-1}\right) dx_{k-1} \end{aligned} \tag{3}$$

但由于积分运算不易实现，粒子滤波采用蒙特卡罗方法将积分运算转换为有限样本点的求和运算。预测之后，根据 k 时刻的观测值及贝叶斯估计对粒子状态进行更新。粒子的状态更新计算如下：

$$\begin{aligned} P\left(x_k \mid y_{1:k}\right) &= \frac{P(\mathrm{y}_k \mid x_k, \mathrm{y}_{1:k-1}) P\left(x_k \mid y_{1:k-1}\right)}{P\left(y_k \mid y_{1:k-1}\right) P\left(y_{1:k-1}\right)} \\ &= \frac{P\left(y_k \mid x_k\right) P\left(x_k \mid y_{1:k-1}\right)}{P\left(y_k \mid y_{1:k-1}\right)} \end{aligned} \tag{4}$$

由以上计算可知，在进行粒子预测和更新的过程中，需要用到所有的观测数据，随着时间的推移，会造成计算量的大大增加。粒子滤波采用序贯重要性采样(SIS)的方式进行递归计算，根据 $k-1$ 时刻的粒子权值计算得到 k 时刻每个粒子的重要性权值如式(5)所示，并进行权值归一化如式(6)所示。

$$w_k^j = w_{k-1}^j \frac{p\left(y_k \mid x_k\right) p\left(x_k \mid x_{k-1}\right)}{q\left(x_k \mid x_{0:k-1}, y_{1:k}\right)} \tag{5}$$

$$w_k^*(\mathrm{j}) = \frac{w_k^j}{\sum_{i=1}^{N} w_k^j} \tag{6}$$

基于上述内容，粒子滤波在 $k=0$ 时刻对所有粒子进行初始化状态信息，随着时间的推移不断地对粒子的状态进行预测和更新，计算所有粒子的权值并归一化，判断是否需要重采样。若需要，则进行重采样操作（复制权重大的粒子，舍弃权重小的粒子）得到一组新的粒子的状态；若不需要，则进入下一时刻的预测更新，实现对目标的循环跟踪。

2　萤火虫算法

群智能优化算法是通过对生物种群的群行为进行数学建模来达到寻优目的的一类算法。萤火虫算法作为群智能优化算法的一种，主要是模仿自然界中萤火虫的发光行为，以个体间的亮度差异为基础进行相互吸引完成寻优过程，达到寻优目的。萤火虫之间的相互吸引主要依靠亮度和吸引度。

2.1　萤火虫的相对亮度

萤火虫的相对亮度是萤火虫之间相互吸引的基础。在相互吸引的过程中，由于萤火虫两两个体的亮度不同，亮度较高萤火虫会吸引亮度较低萤火虫使萤火虫种群一致向最亮萤火虫移动来完成寻优过程。相对亮度公式如下：

$$I_i = I_0 e^{-\gamma r_{ij}^2} \tag{7}$$

其中：I_i 为萤火虫 i 的相对亮度大小；I_0 为最亮萤火虫的亮度，由最优萤火虫所处位置的目标函数值得到；γ 为光强吸收系数，主要用来体现亮度随距离的增加而减弱的特性，一般取 $\gamma \in [0.01, 100]$；r_{ij} 为萤火虫 i 到萤火虫 j 的笛卡儿距离。

2.2　萤火虫的吸引度

在萤火虫相互吸引的过程中，通过吸引度来衡量萤火虫个体吸引力的大小，结合其个体的相对亮度，完成位置更新。吸引度公式如下：

$$\beta_{ij} = \beta_0 e^{-\gamma r_{ij}^2} \tag{8}$$

其中：β_0 为最大吸引度，即在光源处萤火虫的吸引度。

2.3　萤火虫的位置更新

萤火虫种群中的个体会在吸引与被吸引的过程更新自身的

位置信息，从而不断地向最亮萤火虫靠拢使种群分布在亮度较高的区域。位置更新公式如下：

$$x_j = x_j + \beta\left(x_i - x_j\right) + \alpha\varepsilon_i \tag{9}$$

假设萤火虫 j 被萤火虫 i 吸引，萤火虫 j 将按如上公式进行位置更新。其中，x_i, x_j 为萤火虫 i, j 的位置；β 为萤火虫 i 对萤火虫 j 的吸引度；α 为常数，一般取 $\alpha \in [0,1]$；ε_i 是由高斯分布、均匀分布或者其他分布得到的随机数。

3　基于改进的萤火虫算法的粒子滤波

3.1　算法描述

本文主要从以下两个方面对算法进行了改进：

a)由于原始的亮度公式计算复杂度较高，并且没有结合粒子最新的观测值信息，这会导致跟踪准确性下降的问题。因此，本文将观测值信息引入到萤火虫算法的亮度公式中来提高跟踪的准确性。

b)考虑到粒子数目增加时，粒子集在进行相互吸引寻优的过程中，计算量将呈指数增长的问题。本文在充分研究萤火虫算法特点的基础上引入一个新的参数—吸引半径。它主要是随粒子的亮度变化，可以有效地缩减粒子交互时的粒子吸引范围。

与文献[13]相比，本文采用融入观测值信息的相对亮度作为萤火虫算法迭代寻优的基础，使得用于跟踪目标的粒子集能够更加集中地分布在高似然函数区域，跟踪更加准确。同时当多数粒子具有高权值时，表示对准确跟踪有贡献的粒子占据了粒子集的绝大部分，而对跟踪无明显意义的小权值粒子数目大大减少，造成参与重采样步骤的粒子数量有了大幅度的缩减，从而减少了算法在重采样过程中的时间损耗，有效地缩减了算法跟踪时的运行时间。

本文算法在进行目标跟踪过程中的粒子描述如图 2 所示。

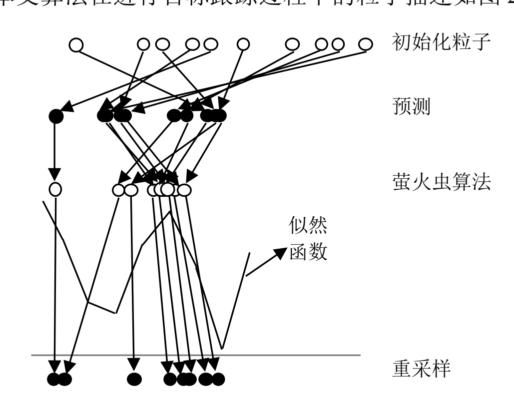


图 2　基于萤火虫算法的粒子滤波的粒子描述图

3.2　基于改进的萤火虫算法的粒子滤波

针对粒子滤波跟踪时所需的粒子数目庞大的问题，基于群智能优化算法的粒子滤波是目前解决该问题的主要方法。但由于粒子群算法需要存储的数据量大，不易更新；而蚁群算法的寻优过程完全依赖于过程中反馈的信息，速度慢，所以本文提出基于自适应吸引半径的萤火虫算法的粒子滤波。

3.2.1 萤火虫相对亮度公式的改进

原始的萤火虫相对亮度公式即式(7)仅依靠最亮粒子信息及粒子间距离来计算相对亮度，若将该相对亮度直接应用于与粒子滤波结合完成粒子的迭代寻优过程，无法实时地利用粒子最新的观测值信息，导致算法的准确性不高。同时，其计算方式的特殊性会在算法运行时带来巨大的计算量。

针对该缺陷，本文采用融入粒子最新的观测值信息的方法来提高算法准确性，减少运行时的计算量。改进后的相对亮度数学表达如下：

$$I = \left(y_k - y\right)^2 \tag{10}$$

其中：I 为改进后的相对亮度；y_k 为粒子的观测值；y 为粒子的预测值。若将相对亮度公式扩充到多维情况中，关键是计算观测值和预测值在多维空间中空间差值的平方。对于二维坐标信息，相对亮度公式应为

$$I = \left(y_k - y\right)^2 + (x_k - x)^2 \tag{11}$$

其中：(x_k, y_k) 是目标观测坐标值；(x, y) 是目标预测坐标值。

相对于原始的相对亮度公式而言，改进后的公式利用每个时刻每个粒子自身的观测值与预测值之间的差值平方来表示粒子每个时刻的相对亮度，不再需要将全部粒子与最优粒子进行比较计算，减少了算法在迭代寻优过程中的计算量。同时，通过加入粒子观测值信息的策略，将算法预测值与观测值的差值实时地反映在相对亮度中，而相对亮度的变化会影响下一时刻粒子的预测值的估计，从而有效地提高了算法跟踪的准确性。

3.2.2 自适应吸引半径的提出

萤火虫算法的寻优过程是对全部萤火虫个体的亮度进行两两相互比较，亮度较低的萤火虫会被亮度较高的萤火虫吸引，并更新自身的位置信息和亮度信息。但若以原始吸引方式进行寻优，随着萤火虫群数目的增加，会造成计算量的巨幅增长，带来很大的时间损耗。因此，本文设计一个新的参数——吸引半径。吸引半径的作用主要是限制萤火虫个体间相互吸引时的吸引范围以及粒子数目。吸引半径的大小由亮度决定，亮度越大的粒子吸引半径越大。

萤火虫亮度与吸引半径的关系如图 3 所示。

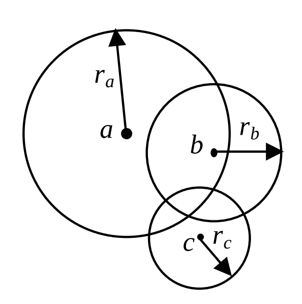


图 3　亮度与吸引半径关系

a,b,c 分别表示三个亮度不同的萤火虫，亮度为 $a > b > c$，显而易见，吸引半径为 $r_a > r_b > r_c$。对于萤火虫 a,b，由于 $r_a > r_{ab}$ 即萤火虫 b 在萤火虫 a 的吸引范围内，所以萤火虫 b 会被萤火虫 a 吸引并更新位置；但对于萤火虫 c，尽管其亮度最小，但由于它不在萤火虫 a,b 的吸引范围内，便不会被萤火虫 a,b 吸引并更新位置。因此，对于两个亮度不同的萤火虫，只有当萤火虫间距离小于较亮萤火虫的吸引半径时才会发生位置更新，否

则不更新位置。

在本文中采用粒子观测值与预测值差值的平方来表示粒子的相对亮度，则相对亮度值越小，表示粒子的预测值和观测值越接近，越靠近最优粒子，粒子的吸引半径则应该越大。设计吸引半径计算公式为

$$r = 10/I \tag{12}$$

其中：r 为粒子的吸引半径；I 为粒子的亮度大小。

基于自适应吸引半径的萤火虫算法在进行粒子间相互吸引时，不再需要将每一个粒子与其他全部粒子进行亮度比较并更新位置，只需将粒子与在其吸引半径内的粒子进行亮度比较完成吸引和位置更新过程即可。这使得算法在迭代寻优过程中参与计算的粒子数目大大降低，从而减少了算法准确跟踪所需的粒子数目，有效地降低了算法的计算复杂度，缩减了算法的运行时间。

3.3 算法步骤

算法 1 基于自适应吸引半径的萤火虫算法的粒子滤波

```
k = 0 时刻初始化
if  k < k_max
for  i < N
    for  j < N
计算 k = k+1 时刻粒子预测值
计算粒子 i, j 的相对亮度及粒子 i 的吸引半径 r
计算粒子 i, j 之间的相互距离 r_ij
if  r_ij < r
        计算粒子 j 的位置更新
end
          j = j+1
        end
        i = i+1
    end
end
更新权值并归一化
if  N_eff ≤ N_th
  重采样
end
状态估计
结束
```

传统粒子滤波算法使用贝叶斯估计理论对粒子集进行更新，本文算法引入了基于自适应吸引半径的萤火虫算法更新粒子集信息。在获得粒子的预测值和观测值信息后，利用萤火虫算法对粒子集进行迭代寻优并完成粒子状态的更新，使粒子集分布在高似然区域，进行更准确的跟踪。

4 实验与分析

4.1 实验条件

实验软硬件环境见表 1。

表 1 实验软硬件环境

硬件环境	软件环境
Intel(R) i5 处理器	MATLAB R2016a
4.00 GB 内存	Windows 10 操作系统

实验采用的系统状态模型和观测模型如下：

$$x(t) = 0.5x(t-1) + \frac{25x(t-1)}{1-[x(t-1)]^2} + 8\cos[1.2(t-1)] + \omega(t) \tag{13}$$

$$y(t) = \frac{x(t)^2}{20} + v(t) \tag{14}$$

该系统为单变量非静态增长模型，是一个典型的非线性系统，传统的粒子滤波方法很难进行准确的跟踪。设观测噪声方差 R=1，滤波时间长度为 50。在萤火虫算法中，一般最大吸引度的设置区间为[0.8,1]，本文设置最大吸引度为 0.8，光强吸收系数设置为 1。

4.2 定量分析

在软硬件条件相同的情况下，对比算法使用标准的粒子滤波算法(以下简称 PF)和文献[14]中提出的萤火虫算法智能优化粒子滤波算法(以下简称 FA-PF)，记录不同算法的运行结果并进行比较和分析。

本文所有结果数据都是通过 100 次重复实验取平均数据值得到的。

4.2.1 精度测试

本文使用均方根误差(又称标准误差，RMSE)公式作为误差大小即预测精度的衡量方式。均方根误差是用来衡量观测值与预测值之间的偏差，能够很好地反映出预测的精密度。均方根误差的计算公式为

$$RMSE = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(x_t - \hat{x}_t)^2} \tag{15}$$

其中：T 为跟踪的时间长度；$\hat{x}_t$ 为状态均值。

根据均方根误差的计算公式可知，状态估计值越接近于观测值，RMSE 的值越小，跟踪准确性较高，算法性能越好，反之则越差。

1）算法性能测试

在不同粒子数目 N=50、N=100、N=200 及不同过程噪声 Q=1、Q=0.1 条件下运行本文算法与 PF、FA-PF 三种算法，并记录不同条件下跟踪的均方根误差值，结果数据见表 2。

表 2 不同粒子数目下几种算法的 RMSE 对比

参数		PF	FA-PF	本文算法	提高率	
					对比 PF	对比 FA-PF
Q=1	N=50	4.9413	4.0585	3.8417	22.3%	5.3%
	N=100	4.3735	3.5640	3.3271	23.9%	6.6%
	N=200	4.0221	3.4576	3.1543	21.6%	8.8%
Q=0.1	N=50	3.3104	3.2162	3.1439	5.1%	2.3%
	N=100	3.2053	3.1507	2.9825	6.9%	5.3%
	N=200	3.0060	2.9693	2.8555	5.0%	3.8%

由表 2 可知，本文算法在不同粒子数目、过程噪声条件下

RMSE 值都较其他两种算法小，说明本文算法的准确性在不同情况下均高于 PF 和 FA-PF 算法。在不同过程噪声下，随着粒子数目 N 的增加，三种算法的准确性都有所提高。

除此之外，Q=1 时，本文算法的准确性在 N=100 时依然高于 FA-PF 算法在 N=200 时；Q=0.1 时，本文算法在 N=100 时的准确性近似于 FA-PF 算法在 N=200 时的准确性，说明本文算法在粒子数目少的情况下依旧能保证跟踪的准确性。

究其原因，由于本文算法在重采样前利用改进后的萤火虫算法对粒子集进行迭代寻优操作，使粒子分布在高似然区域粒子的权值都较高，进行准确跟踪所需的粒子数目也有所下降。

通过表 2 数据计算可得，在不同的过程噪声下，本文算法相对于 PF 以及 FA-PF 的平均提高率如表 3 所示。

表 3　不同过程噪声下本文算法准确性的平均提高率

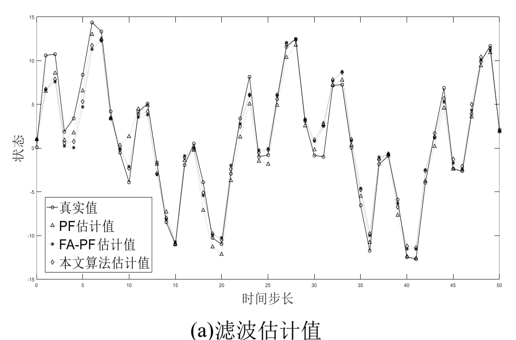
参数	平均提高率	
	对比 PF	对比 FA-PF
Q=1	22.6%	7%
Q=0.1	5.7%	3.8%

由表 3 数据可知，Q=1 时，本文算法准确率较最新的 FA-PF 算法提升 7%，较 PF 算法有 22.6%的提升；Q=0.1 时，本文算法的准确性较 FA-PF 和 PF 有 3.8%和 5.7%的提高。这是由于本文算法为确保滤波器的精度，在萤火虫的迭代寻优过程中引入了最新的观测值信息。

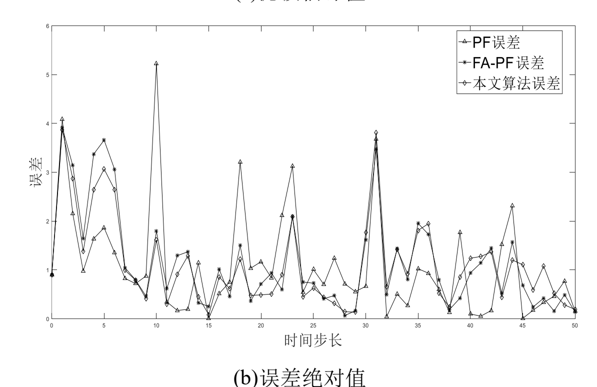
2）算法跟踪效果测试

在相同过程噪声，不同粒子数目 N=50、N=100 条件下运行本文算法与 PF、FA-PF 算法并记录三种不同算法跟踪的状态估计值及误差绝对值数据，结果如图 4、5 所示。

a)当 N=50，Q=1 时，实验的估计值及误差绝对值分别如图 4(a) (b)所示。



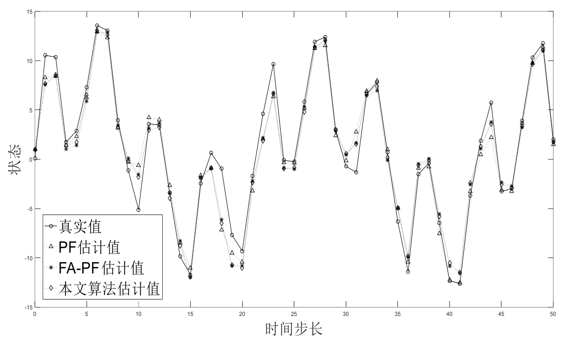
(a)滤波估计值



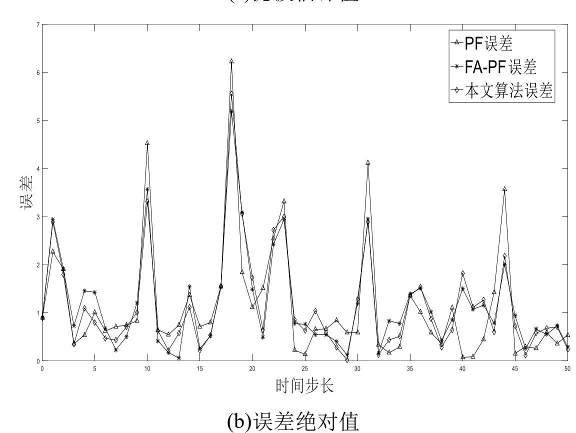
(b)误差绝对值

图 4　N=50 时本文算法的跟踪结果与其他算法对比

b)当 N=100，Q=1 时，实验的估计值及误差绝对值分别如图 5(a) (b)所示。



(a)滤波估计值



(b)误差绝对值

图 5　N=100 时本文算法的跟踪结果与其他算法对比

从图 4(a)、5(a)可以看出，本文算法在不同的粒子数目下的滤波估计值要比 PF 及 FA-PF 算法更加接近于真实值，反映在图 4(b)、5(b)中即本文算法的误差绝对值要小于 PF 及 FA-PF 算法，本文算法的准确性在粒子数目不同的情况下均高于 PF 及 FA-PF 算法。

这主要是由于本文算法在萤火虫算法中采用了融合粒子观测值信息的相对亮度计算方式，在粒子集的迭代寻优过程中根据相对亮度更加准确地调整粒子的位置信息，这使得粒子更加准确地分布在其真实值附近，从而算法的准确性得到了一定的提高。

4.2.2 实时性测试

在软硬件条件相同的条件下，在不同粒子数目、不同过程噪声条件下运行不同算法并记录算法运行时间进行对比，结果对比见表 4。

表 4　不同粒子数目下几种算法运行时间对比

参数		PF	FA-PF	本文算法	对比 FA-PF 缩减率
	N=50	0.0101	0.0264	0.0227	14.02%
Q=1	N=100	0.0175	0.0629	0.0585	7.00%
	N=200	0.0404	0.1685	0.1515	10.09%
	N=50	0.0093	0.0194	0.0177	8.7%
Q=0.1	N=100	0.0152	0.0457	0.0431	5.7%
	N=200	0.0323	0.01241	0.1241	8.8%

从表 4 可以看出，在不同粒子数目、不同过程噪声条件下，本文算法的运行时间较 PF 算法存在一些延长，但结合表 2 数

据，本文算法的准确性较 PF 算法有大幅度的提高。这是由于本文算法在重采样前使用了粒子集的迭代寻优操作，使多数粒子的权值都较高，所以本文以运行时间加长为代价换来准确性的提高。并且在不同过程噪声条件下，三种算法的运行时间随着粒子数目的增加都有一定的延长。

通过表 4 数据计算可得在不同的过程噪声下，本文算法的运行时间相对于 FA-PF 的平均缩减率如表 5 所示。

表 5　不同过程噪声下，本文算法运行时间的平均缩减率

参数	平均缩减率
Q=1	10.4%
Q=0.1	7.7%

由表 4 和 5 中数据可以看出，在不同粒子数目以及不同过程噪声的情况下，本文算法的运行时间较 FA-PF 算法都存在一定程度的缩减。在 Q=1 及 Q=0.1 时，本文算法的运行时间对比 FA-PF 算法有 10.4%和 7.7%的平均缩减。

这是由于在使用萤火虫算法进行粒子间的相互吸引完成迭代寻优过程中，本文提出并使用了自适应吸引半径来控制粒子的吸引范围，从而大大减少了参与计算的粒子数目，使得算法的运行时间有了进一步的缩减。

4.3　定性分析

为了直观地体现出本文算法的跟踪性能，在主流的视频集 Visual Tracking Benchmark[20]上选取视频进行实验。在跟踪过程中，采用颜色直方图作为跟踪特征，取粒子数 N=100。将本文算法与 PF 算法的跟踪效果进行对比，并对跟踪结果的部分帧进行截图，如图 6~9 所示。

在 Coke 视频中，跟踪场景较为简单，跟踪目标的形态也并未发生大幅度变化，跟踪目标作无规律的圆周运动，是典型的非线性系统下的目标跟踪。图 6 为视频序列的完整场景图，图 7 为 PF 算法跟踪视频序列结果，图 8 为文献[14]中提出的的 FA-PF 算法跟踪视频序列结果，图 9 为本文算法跟踪视频序列结果。



图 6　Coke 视频完整场景图

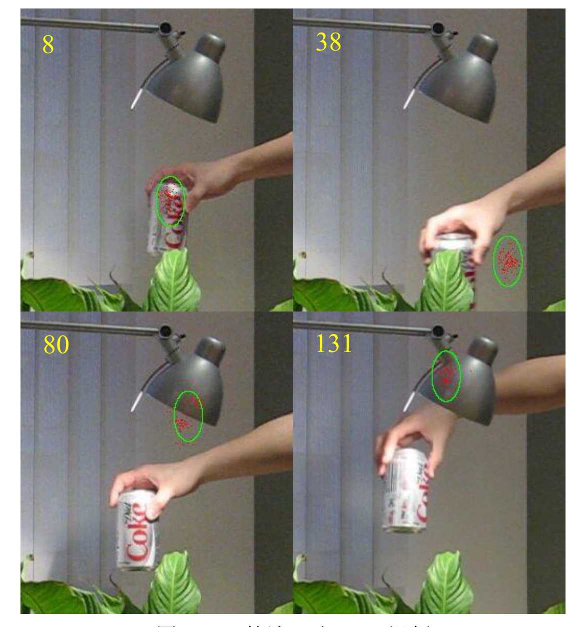


图 7　PF 算法跟踪 Coke 视频

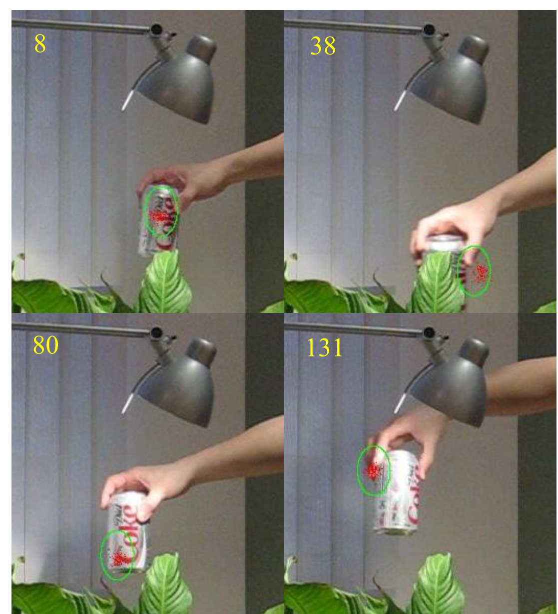


图 8　FA-PF 算法跟踪 Coke 视频

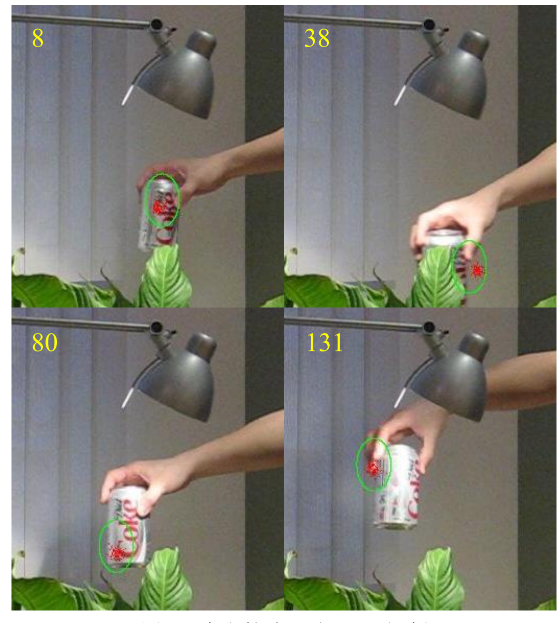


图 9　本文算法跟踪 Coke 视频

由图 7~9 可明显地看出，PF 算法的粒子分布很散乱，粒子分布不均匀使得对跟踪有意义的粒子数目大大减少，这是导致 PF 算法跟踪失败的重要原因。FA-PF 算法的粒子分布较 PF 算法更为集中，本文算法的粒子分布最为集中，主要集中在跟踪边界框的中心位置，这是由于本文算法利用改进的萤火虫算法对粒子集进行了迭代寻优操作。

在第 38 帧，跟踪目标运动到场景的后方发生了部分遮挡；在第 80 帧和第 131 帧，由于散乱的粒子无法集中到高似然区域，这些原因都导致 PF 算法跟踪失败。相比于基本实现准确跟踪的 FA-PF 算法而言，本文算法使用改进的萤火虫算法有效地对全部粒子进行迭代寻优，使粒子集尽可能多地分布在目标附近，表示跟踪有贡献的粒子数量较大，从而实现更加准确的跟踪。

除此之外，本文算法由于引入了自适应吸引半径，有效控制了参与寻优的粒子数目，缩减了迭代寻优过程。由表 4 数据可知，算法运行的时间也有了 7%的缩减。

5 结束语

粒子滤波算法需要大量的粒子才能保证算法的准确性，随着权值的不断削减，用于跟踪的粒子数目越来越少，需要对粒子集进行重新采样。但这会严重削减粒子的多样性，甚至导致最终只剩下一种粒子，导致跟踪失败。本文使用改进的萤火虫算法对粒子滤波算法进行优化。

根据两种算法的特点，对萤火虫的亮度公式进行了改进，结合粒子的观测值信息使粒子集分布在全局最优的区域，保证了算法的准确性；同时提出了自适应吸引半径参数，避免了粒子数目增加时计算量的大幅增长，降低了算法的运算复杂度。定量分析和定性分析均表明，本文算法有效地减少了跟踪过程中所需的粒子数目，在粒子数目较少的条件下依然能够实现准确的跟踪，同时本文算法的运行时间也相对于 FAPF 有了一定的缩减。

参考文献：

[1] Yan Yue, Wang Jingling, Li Chuanzhen, *et al.* Object tracking using SIFT features in a particle filter [C]// Proc of IEEE International Conference on Communication Software and Networks. Piscataway, NJ: IEEE Press, 2011: 384-388.

[2] 王宇霞，赵清杰，蔡艺明，等. 基于自重构粒子滤波算法的目标跟踪 [J]. 计算机学报, 2016, 39 (7): 1294-1406. (Wang Yuxia, Zhao Qingjie, Cai Yiming, *et al.* Tracking by auto-reconstructing particle filter trackers [J]. Chinese Journal of Computers, 2016, 39 (7): 1294-1406.)

[3] Khan M W, Salman N, Ali A, *et al.* A comparative study of target tracking with Kalman filter, extended Kalman filter and particle filter using received signal strength measurements [C]// Proc of IEEE International Conference on Emerging Technologies. Piscataway, NJ: IEEE Press, 2016: 1-6.

[4] 林庆，王新. 基于改进的粒子滤波算法的目标跟踪 [J]. 信息技术, 2017 (10): 88-92. (Lin Qing, Wang Xin. Improved particle filter algorithm based target tracking [J]. Information Technology, 2017 (10): 88-92.)

[5] Kwok N M, Fang Gu, Zhou Weizhen, *et al.* Evolutionary particle filter: re-sampling from the genetic algorithm perspective [C]// Proc of IEEE//RSJ International Conference on Intelligent Robots and Systems. Piscataway, NJ: IEEE Press, 2016: 2935-2940.

[6] 鲍丙计. 基于粒子滤波的视频目标跟踪算法研究 [D]. 合肥: 安徽大学, 2016. (Bao Bingji. Research of video target tracking algorithm based on particle filter [D]. Hefei: Anhui University, 2016.)

[7] Wu Yanhai, Xie Xiamin, Han Zishuo. Research on target tracking algorithm based on particle filter and Mean-Shift [J]. Applied Mechanics & Materials, 2014, 457-458: 1050-1053.

[8] 张才千，葛磊，韩东. 基于目标跟踪的粒子群粒子滤波算法研究 [J]. 计算机仿真, 2014, 31 (8): 392-396. (Zhang Caiqian, Ge Lei, Han Dong. Research on particle swarm particle filter algorithm based on target tracking [J]. Computer Simulation, 2014, 31 (8): 392-396.)

[9] Wang Fasheng, Lin Baowei, Li Xucheng. An ant particle filter for visual tracking [C]// Proc of IEEE//ACIS International Conference on Computer and Information Science. Piscataway, NJ: IEEE Press, 2017: 417-422.

[10] Herisn S M K, Khaloozadeh H. Ant colony estimator: an intelligent particle filter based on ACOR [J]. Engineering Applications of Artificial Intelligence, 2014 (28): 78-85.

[11] Wang W, Tan Q K, Chen J, *et al.* Particle filter based on improved genetic algorithm resampling [C]// Proc of Guidance, Navigation and Control Conference. Piscataway, NJ: IEEE Press, 2017: 346-350.

[12] 朱文超，许德章. 一种基于人工萤火虫群优化的改进粒子滤波算法 [J]. 计算机应用研究, 2014, 31 (10): 2920-2924. (Zhu Wenchao, Xu Dezhang. Improved particle filter algorithm based on artificial glowworm swarm optimization [J]. Application Research of Computers, 2014, 31 (10): 2920-2924.)

[13] Gao Mingliang, Li Lili, Sun Xianming, *et al.* Firefly algorithm (FA) based particle filter method for visual tracking [J]. Optik-International Journal for Light and Electron Optics, 2015, 126 (18): 1705-1711.

[14] 田梦楚，薄煜明，陈志敏，等. 萤火虫算法智能优化粒子滤波 [J]. 自动化学报, 2016, 42 (1): 89-97. (Tian Mengchu, Bo Yuming, Chen Zhimin, *et al.* Firefly algorithm intelligence optimized particle filter [J]. Acta Automatica Sinica, 2016, 42 (1): 89-97.)

[15] 王法胜，鲁明羽，赵清杰，等. 粒子滤波算法 [J]. 计算机学报, 2014, 37 (8): 1679-1694. (Wang Fasheng, Lu Mingyu, Zhao Qingjie, *et al.* Particle filtering algorithm [J]. Chinese Journal of Computers, 2014, 37 (8): 1679-1694.)

[16] Zhou Tianrun, Ouyang Yini, Wang Rui, *et al.* Particle filter based on real-time compressive tracking [C]// Proc of International Conference on Audio. Piscataway, NJ: IEEE Press, 2017: 754-759.

[17] Gonzalez M, Collet C. Robust body parts tracking using particle filter and dynamic template [C]// Proc of IEEE International Conference on Image

Processing. Piscataway, NJ: IEEE Press, 2011: 529-532.

[18] Nobahari H, Raoufi M, Sharifi A. A heuristic filter based on firefly algorithm for nonlinear state estimation [C]// Proc of IEEE Symposium Series on Computational Intelligence. Piscataway, NJ: IEEE Press, 2017.

[19] Su Yingya, Zhao Qingjie, Zhao Liujun, *et al.* Abrupt motion tracking using a visual saliency embedded particle filter [J]. Pattern Recognition, 2014, 47 (5): 1826-1834.

[20] Wu Yi, Lim Jongwoo, Yang Ming-hsuan. Online object tracking: a benchmark [C]// Proc of Computer Vision and Pattern Recognition. Piscataway, NJ: IEEE Press, 2013: 2411-2418.